An Introduce to Computer Vision and Video Surveillance System

Phạm Duy Tùng,
Faculty of Applied Mathematics and Infomatics, Hanoi University of Technology.

Tóm tắt nội dung.

Trong bài viết này tôi sẽ giới thiệu về lĩnh vực Computer Vision, ứng dụng của nó và xu hướng nghiên cứu qua việc trình bày 3 thuật toán quan trọng nhất trong computer vision và chúng được sử dụng trong hầu hết các hệ thống Computer Vision hiện nay đó là: (1) các thuật toán cho mô hình Gaussian Mixture, (2) particle filtering (condensation filtering), (3) optical flow/motion estimation. Cuối cùng tôi trình bày về hệ thống Video Surveillance, qua đó các bạn có thể xây dựng hệ thống mới cho các mục đích khác.

1. Giới thiệu.

Ngày nay với sự phát triển mạnh mẽ của công nghệ sản xuất chip, dẫn đến sự ra đời của những hệ thống máy tinh vô cùng mạnh mẽ với giá rất thấp. Bên cạnh đó các thuật toán cũng ngày càng được cải tiến về tốc độ tính toán cũng như độ chính xác. Cùng với sự bùng nổ của dữ liệu số chúng ta cần các ứng dụng thông minh hơn, mạnh mẽ hơn giúp ta tổ chức, quản lý dữ liệu một cách hiệu quả. Các từ khóa như: "Recognition", "Minning", "Synthesis" (RMS) sẽ trở lên phổ biến trong thời gian tới. Trong những năm gần đây lĩnh vực "Video minning" hay "Computer Vision" phát triển mạnh mẽ, có ứng dụng quan trọng trong y tế, an ninh, robot, giải trí…Đặc biệt là các ứng dụng thời gian thực (Real-time System). Sự phát triển của các ứng dụng này sẽ làm thay đổi hoàn toàn cách chúng ta tương tác với máy tính, cách chúng ta làm việc, giải trí, chơi game…

Những ứng dụng của Computer Vision.

Các thiết bị quay phim kỹ thuật số đã trở lên phổ biến trong nhưng năm gần đây. Chúng được đặt ở văn phòng, bệnh viện, lớp học, ... Một khối lượng dữ liệu cực lớn được thu bởi những "*con mắt điện tử*" này. Các công nghệ Computer Vision sẽ mang lại trí thông minh cho những "*con mắt điện tử*" này bằng cách đưa thêm vào một "*bộ não điện tử*". Với cả đôi mắt và bộ não, CV sẽ trở lên vô cùng hữu ích. Nó có thể được sử dụng trong Video giám sát, giải trí,

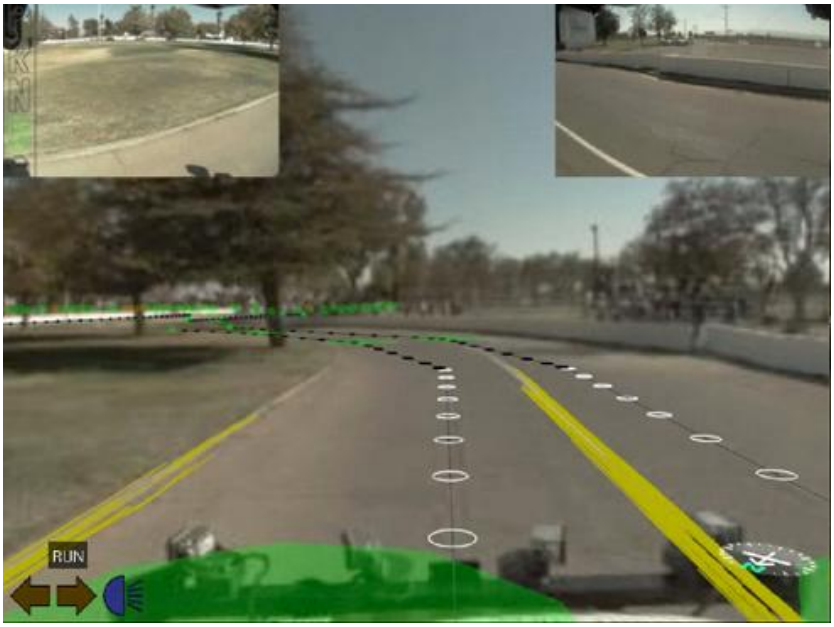
phương tiện giao thông tự lái, và hệ thống trợ giúp điều khiển, khoa hoc robot, chăm sóc sức khỏe thông minh...Với sự lớn mạnh nhanh chóng của các công nghệ Computer Vision tôi tin vào tương lài tươi sáng của lĩnh vực này. Bây giờ chúng ta sẽ tìm hiểu một số ứng dụng chính của CV.

Video Surveillance/Security

Một hệ thống Video surveillance hoàn chỉnh bao gồm: tách Foreground(vùng ảnh thay đổi), phát hiện đối tượng, theo dõi đối tượng, phân tích đối tượng, phân tích hành vi của đối tượng.

Hệ thống Video Surveillance truyền thống đã được sử dụng rộng rãi trong sân bay, ngân hàng, bãi đỗ xe...Tuy nhiên để có được thông tin hữu ích thì con ngươi phải "*gián mắt*" vào màn hình với sự tập trung cao để có thể phát hiện ra những hiện tượng bất thường. Một hệ thống Video surveillance tự động bằng cách đưa vào những thuật toán CV giúp giải phóng sức lao động, công việc giám sát sẽ thú vị hơn rất nhiều. Hệ thống Video surveilance sẽ được giới thiệu trong phần sau.

Phương tiện giao thông tự lái và các hệ thống hỗ trợ điều khiển.



Lái xe an toàn là một vấn đề rất quan trọng trong cuộc sống của chúng ta. CV có thể đóng vai trò như là một con mắt thứ 3 cho người lái xe giúp nâng cao tính an toàn của các phương tiện giao thông. Ví dụ về việc sử dụng CV cho phương tiện giao thông thông minh như là: hỗ trợ đỗ xe, nhận diện các mốc ranh giới trên đường giúp ôtô đi đúng đường, phát hiện và nhân dạng các tín hiệu giao thông giúp cảnh báo lái xe, phát hiện chướng ngại vật đặc biệt là người đi bộ dưới lòng đường...

Entertainment.

Một ứng dụng thú vị khác của CV là làm tăng tính thực tế, nó kết hợp với đọc nội dung video, tách đối tượng trong video, biểu diễn lại đối tượng bằng các mô hình đồ họa. Sau đó đối tượng 3D ảo này được nhúng trong Video. Các ứng dụng này có thể ứng dùng để thiết kế những game thú vị hơn, ngươi chơi có được những trải nghiệm thực tế hơn.

Smart Health Care

CV có thể được sử dụng trong chăm sóc sức khỏe cho người già, và người tàn tật. Theo dõi con người và phân tích hành vi có thể giúp phát hiện những hiện tượng bất thường như một người bị ngã. Con người cũng có thể sử dụng tay hay ngôn ngữ cơ thể để giao tiếp với máy tính, ý tưởng này có thể giúp xây dựng văn phòng thông minh hay ngôi nhà thông minh.

2. Những thuật toán quan trọng lĩnh vực Computer Vision (CV).

CV algorithms có thể được chia làm hai nhóm:

- các kỹ thuật xử lý ảnh ở mức thấp.*(low-level image-processing techniques).*

-Các kỹ thuật phân tích ở mức cao(*high-level analysis techniques).*

Các kỹ thuật xử lý ảnh ở mức thấp bao gồm: lọc nhiễu và tách các đặc trưng được nghiên cứu rộng rãi, các bạn có thể tìm thấy trong các tài liệu về xử lý ảnh. Trong khuôn khổ bài viết này tôi chỉ trình bày các thuật toán phân tích ở mức cao. Các kỹ thuật phân tích mức cao thường được tiếp cận qua các mô hình xác suất. Chúng ta sẽ tìm hiểu về mô hình Gaussian Mixture và particle filters vì chúng được xử dụng cực kỳ phổ biến trong phân tích và theo dõi chuyển động.

Hiện naycác hướng nghiên cứu trong CV chủ yếu tập trung vào các thuật toán phân tích ở mức cao, chúng chủ yếu dựa trên các mô hình xác suất.

2.1 Gaussian Mixture Model

Để có thể theo dõi được các đối tượng , những mô tả đặc điểm của đối tượng là rất quan trọng. Những đặc điểm của đối tượng sẽ được mô tả thông qua các phân phối thống kê của vùng quan sát (được chia thành foreground object hoặc là background). Mô hình Gaussian Mixture được sử dùng rộng rãi để mô tả vùng quan sát và là một phần không thể thiếu của các thuật toán nhận dạng, theo doi trong CV.

Mô hình Gaussian Mixture với m thành phần được mô tả như là tổng của m phân phối xác suất Gaussian:

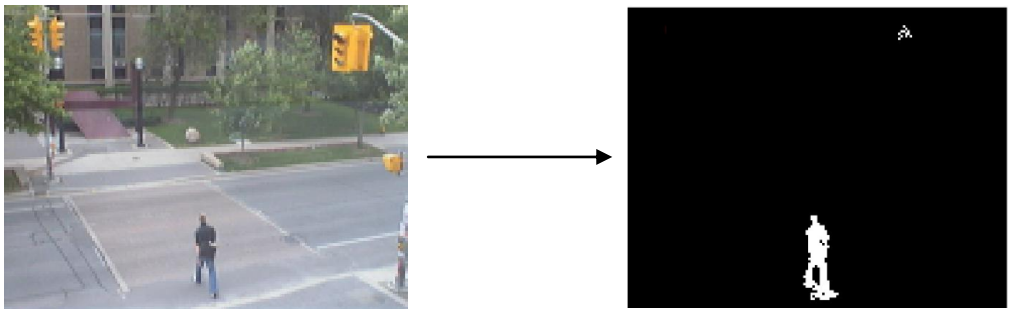
$$P(x) = \sum_{i=1}^{m} \pi_i N(x, \mu_i, \delta_i^2 I)$$

Trong đó μ_i, δ_i lần lượt là kỳ vọng và phương sai của thành phần thứ i, I là ma trận đơn vị và π_i là trọng số của thành phần thứ i sao cho $\sum_{i=1}^{m} \pi_i = 1$

Bằng cách sử dụng thuật toán Estimation-Maximization (EM) ta có thể ước lượng được tham số cho mô hình.

Adaptive background mixtrure models for real-time tracking

Việc tách Foregound ra khỏi Backround là bước đầu tiên, đóng vai trò quyết định đến độ chính xác của hầu hết các hệ thông CV hiện nay. FG có thể hiểu là miền chuyển động, còn BG có thể hiểu là miền tĩnh trong vùng quan sát.



Thuật toán *Adaptive background mixture models for real-time tracking* được sử dụng rất phổ biến để tách FG vì có độ chính xác và tốc độ cao. Thuật toán có khả năng thích nghi cao với sự thay đổi của môi trường.

Phần này mình viết dựa trên bài báo "*Adaptive background mixture models for real-time tracking"* của Chris Stauffer và W.E.L Grimson.

Phương pháp truyền thống để tách FG là lấy trung bình các bức ảnh trong một khoảng thời gian để xấp xỉ BG. Phương pháp này khá hiệu quả nếu các đối tượng chuyển động liên tục và BG xuất hiện trong một khoảng thời gian đáng kể. Nó sẽ không còn hiệu quả nếu có nhiều đối tượng chuyển động và đặc biệt là khi chúng chuyển động chậm. Sự thay đổi ánh sáng của môi trường cũng gây ra rất nhiều vấn đề cho các phương pháp tách FG. Thay vì chỉ ra mô hình cụ thể cho tất cả các giá trị của các pixel đều có một phân phối xác xuất xác định, chúng ta xây dựng mô hình mô tả các giá trị của các pixel như là một mô hình *Gaussian Mixture*. Dựa trên sự ổn định và sự biến đổi của mỗi thành phần (Gaussian) của mô hình mixture, chúng ta sẽ chỉ ra thành phần nào tương ứng

với background colors. Giá trị của pixel nào không phù hợp với phân phối của backgound sẽ được coi là FG.

Hệ thống này rất hiệu quả trong điều kiện ánh sáng thay đổi, các chuyển động lặp lại, các đối tượng chuyển động chậm, và viêc đưa vào hay loại đi các đối tượng trong vùng quan sát.

Phương pháp này có hai tham số quan trọng là: *α-hằng số học (learning constant)* và *T- xác suất nhỏ nhất mà BG xuất hiện trong miền quan sát.* Không cần thay đổi các tham số này hệ thống làm việc hiệu quả cả trong nhà lẫn ngoài trời.

Trong phương pháp này ta coi giá trị của một pixel cụ thể biến đổi theo thời gian như là một "*pixel process*". Một pixel process là một chuỗi thời gian của các giá trị pixel, e.g. giá trị thực cho ảnh xám và vector cho ảnh màu. Các giá trị của một pixel cụ thể cho đến thời điểm t bất kỳ được biểu diễn như sau:

$$\{X_1, X_2, \dots, X_t\} = \{I(x_0, y_0, i): 1 \leq i \leq t\} \qquad (1)$$

Trong đó I là một chuỗi các ảnh.

$\{X_1, X_2, \dots, X_t\}$ - *recent history* của mỗi pixel được mô tả bởi K Gaussion distributions mixture . Xác suất của quan sát giá trị pixel hiện tại là

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \qquad (2)$$

Trong đó K là số phân phối, $\omega_{i,t}$ là trọng số của thành phần Gaussian thứ I trong mô hình tại thời điểm t(hay nói cách khác nó là xác xuất để X_t có phân phối $\eta(X_t, \mu_{i,t}, \Sigma_{i,t})$), $\mu_{i,t}$ là giá trị trung bình của thành phần Gaussian thứ I trong mô hình tại thời điểm t, $\Sigma_{i,t}$ là ma trận hiệp phương sai của thành phần Gaussian thứ I trong mô hình tại thời điểm t, và η là hàm mật độ xác suất Gaussian

$$\eta(X_t, \mu_{i,t}, \Sigma_{i,t}) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{\frac{1}{2}(X_{t-} \mu_t)^T \Sigma^{-1} (X_{t-} \mu_t)} \qquad (3)$$

K được xác định tùy thuộc vào dung lương bộ nhớ hiên có và sức mạnh tính toán của hệ thống, thông thường K từ 3 cho đến 5. Cũng vì lý do tính toán, ma trận hiệp phương sai $\Sigma_{i,t}$ giả thiết có dạng

$$\Sigma_{i,t} = \sigma_k^2 I \qquad (4)$$

Với giả thiết này thì red, green, and blue pixel values được coi là độc lập và có cùng phương sai. Mặc dù trong thực tế điều này không chính xác lắm nhưng để tránh phức tạp trong việc tính ma trận nghịch đảo ta chấp nhận giả thiết này.

Bởi vì mọi pixel trong image đều được mô tả bởi một một Gaussian mixture, nên việc cài đặt thuật toán *Expectation-Maximization (EM)* trên một vùng dữ liệu gần đây sẽ có độ phức tạp tính toán cao. Thay vào đó, ta sẽ cài đặt thuật toán *on-line K means approximation*. Mỗi giá trị pixel mới sẽ được kiểm tra lại với K phân phối Gaussian, cho đến khi tìm được phân phối phù hợp. Một giá trị pixel được gọi là phù hợp với một phân phối Gaussian nếu giá trị đó không vượt quá 2.5 lần độ lệch tiêu chuẩn của phân phối đó.

Nếu giá trị pixel không phù hợp với bất kỳ phân phối nào của mô hình mixture thì phân phối kém phù hợp nhất được thay thế bằng một phân phối mới, trong đó giá trị pixel hiện tại là giá trị trung bình của của phân phối mới, phương sai của phân phối được khởi tạo với một giá trị đủ lớn, trọng số của phân phối được gán một giá trị nhỏ.

Các trọng số của K phân phối tại thời điểm t sẽ được hiệu chỉnh như sau:

$$\omega_{k,t} = (1-\alpha)\ \omega_{k,t-1} + \alpha(M_{k,t}) \qquad (5)$$

Trong đó α là hằng số và ta gọi đó là *hệ số tốc độ học*, $M_{k,t}$ = 1 nếu giá trị pixel phù hợp với phân phối thứ k, bằng 0 nếu ngược lại. Giá trị 1/ α định nghĩa hằng số thời gian nó thể hiện tốc độ thay đổi của các tham số của các phân phối. $\omega_{k,t}$ là xác suất để giá trị pixel tại thời điểm t phù hợp với phân phối thứ k trong điều kiện đã quan sát được các giá trị pixel đến thời điểm t-1.

Các tham số μ, σ cho phân phối không phù hợp được giữ nguyên, các tham số của phân phối phù hợp với quan sát mới (giá trị pixel mới) được update như sau:

$$\mu_t = (1-\rho)\mu_{t-1} + \rho X_t \qquad (6)$$

$$\sigma_t^2 = (1-\rho)\ \sigma_{t-1}^2 + \rho(X_t - \mu_t)^T(X_t - \mu_t) \qquad (7)$$

Trong đó ρ *là hệ số tốc độ học* thứ hai được xác định như sau:

$$\rho = \alpha\, \eta(X_t | \mu_k, \sigma_k) \qquad (8)$$

Mỗi khi những tham số của mô hình mixture của mỗi pixel thay đổi, ta muốn chỉ ra những thành phần nào của mô hình được sinh ra bởi background. Để làm

được điều này, trước tiên chúng ta sắp xếp K thành phần của mô hình theo chiều giảm của giá trị $\frac{\omega_{k,t}}{\sigma_{k,t}}$. Để hiểu lý do tại sao ta làm như vậy, các bạn để ý rằng nếu một pixel là backround thì nó sẽ phù hợp với một thành phần nào đó của của mô hình mixture, theo cách update trọng số ta có $\omega_{k,t}$= *(1-α)* $\omega_{k,t-1}$ + α dễ thấy $\omega_{k,t} > \omega_{k,t-1}$. Thêm nữa giá trị của pixel do background sinh ra ổn định, ít biến đổi hay nói cách khác quan sát do background sinh ra sẽ có phươn sai nhỏ. Vì những lý do đó tỉ số $\frac{\omega_{k,t}}{\sigma_{k,t}}$ có nói lên độ phù hợp với mô hình background của giá trị pixel vừa quan sát. Ngược lại, khi pixel là foreground, nói chung nó không phù hợp với K thành phần của mô hình, dẫn đến phải tạo ra một thành phần mới hoặc làm tăng phương sai của phân phối hiện tại.

Sau đó B thành phần đầu tiên sẽ được coi là mô hình của background, B được tính như sau:

$$B = argmin_b(\sum_{k=1}^{b} \omega_{k,t} > T) \qquad (9)$$

Trong đó T là tỉ số cực tiểu của mô hình background. Nói cách khác T là xác suất nhỏ nhất mà background xuất hiện trong vùng quan sát.

Bằng phương pháp này chúng ta có thể tách FG ra khỏi BG khá chính xác, tốc độ nhanh lên được sử dụng rất rộng rãi trong các ứng dụng thời gian thực.

Tài liệu tham khảo:

[1] R. Hammond and R. Mohr, "Mixture densities for video objects recognition" 2000 IEEE International conference on pattern Recognition, pp, 71-75.

[2] R. Wilson, "MGMM: multiresolution Gaussian Mixture Models for computer vision" 2000 IEEE International conference on Pattern Recognition, pp 212-215.

[3] M. Harville, G. Gordon and J. Woodfill, "Foreground segmentation using adaptive mixture model in color and depth", 2001 IEEE Workshop on Detection and Recognition of Events in Video, pp 3-11.

[4] Y. Zhu and K. Fujimura, "Driver face tracking using Gaussian mixture model (GMM)", IEEE Intelligent Vehicles Sysposium, Jun 2003, pp 587-592.

[5] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction", ICPR 2004, pp 28-31.

[6] C. Stauffer, W.E.L Grimson, "Adaptive background mixture models for real-time tracking", 1999 IEEE CVPR.

[7] N. Friedman, S. Russell, "Image segmentation in video sequences: A probabistic approach", Computer Science Division, University of California.

2.2 Particle Filter/Condensation Filter

(vì thời gian có hạn phần này mình chưa đọc kịp mong các bạn thông cảm, mình sẽ trình bày vào dịp khác)

Các bạn có thể tìm đọc thuật toán này trong một số tài liêu:

[8] M.S Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," IEEE Trans. Signal Processing, 50(2), Feb. 2002, pp, 174-188.

[9] F. Gustafsson, F. Gunnarsson, N. Bergman, U.Forssell, J. Jansson, R. Kalsson, and P.J. Nordlund, "Particle filter for positioning, navigation, and tracking," IEEE Trans, On Signal processing, 50(2), Feb, 2002, pp, 425-437.

[10] P.M. Djuric, J.H. Kotecha, J. Zhang, Y. Huang, T.Ghirmai, M.F. Bugallo, and J. Miguez, "Particle filtering," IEEE Signal Processing Magazine, Sept, 2003, pp, 19-38.

2.3 Optical Flow/Motion Estimation.

Optical flow estimation là được sử dụng để tính toán trường chuyển động(motion field) trên màn hình. Bên cạch việc theo dõi các đối tượng, trường chuyển động cũng rất quan trọng khi theo dõi vật chuyển động trong video.

Lucas-Kanade method

Lucas-Kanade method là một trong những phương pháp hiệu quả để tính toán vùng chuyển động. Nội dung phương pháp được tóm tắt như sau:

Một số điều kiện của thuật toán :

1.cường độ sáng của pixel không đổi theo t,

2. chuyển động nhỏ,

3. các pixel lân cận cũng chuyển động cùng vận tốc.

Tại thời điểm t pixel(x, y) có giá trị màu là I(x, y, t), sau thời gian δt vị trí của pixel này là *(x+δx,y+δy) và I(x, y, t) = I(x+Δδx,y+δy,t+δt)* (vì có đk 1)

Mà $I(x+\Delta\delta x,y+\delta y,t+\delta t) \approx I(x, y, t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t$

Suy ra : $\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t = 0$

Hay $\frac{\partial I}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial I}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial I}{\partial t} = 0$

Đặt $V_x = \frac{\partial x}{\partial t}$ $V_y = \frac{\partial y}{\partial t}$ ta có : $\frac{\partial I}{\partial x}V_x + \frac{\partial I}{\partial y}V_y + \frac{\partial I}{\partial t} = 0$ (*)

Phương trình (*) là phương trình bậc nhất hai ẩn V_x, V_y.

Với điều kiện 3 ta có thể sử dụng các giá trị pixel lân cận để giải phương trình trên.

Khi đó ta có hệ m phương trình với m>=2

$$\begin{bmatrix} \frac{\partial I(p1)}{\partial x} & \frac{\partial I(p1)}{\partial y} \\ \vdots & \vdots \\ \frac{\partial I(pm)}{\partial x} & \frac{\partial I(pm)}{\partial y} \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \frac{\partial I(p1)}{\partial t} \\ \vdots \\ \frac{\partial I(pm)}{\partial t} \end{bmatrix}$$

Giải hệ quá xác định này bằng phương pháp *bình phương cực tiểu(Least-Quares)* ta được vector vận tốc của đối tượng chuyển động.

Tài liệu tham khảo :

[11]J. Barron, D. Fleet, and S. Beauchemin, "Performance of optical flow techniques", Int. J. Comput.Vis., vol 12, no. 1, 1994, pp. 42-77.

[12] A. Verri and T. Poggio, "Motion field and optical flow: qualitative properties," IEEE Trans. On PAMI, 11(5), May 1999, pp.490-498.

[13]A. Giachetti, M. Campani, and V. Torre, "The use of optical flow for road navigation," IEEE Trans. On Robotics and Automation, 14(1), Feb. 1998, pp. 34-48.

3. Giới thiệu hệ thống Camera giám sát.

Các hệ thống giám sát qua video truyền thống thường tốn công sức mà hiệu quả không cao. Giám sát qua Video kết hợp với Computer vision techniques, giúp tiết kiệm lao động và tăng độ chính xác của hệ thống. Đầu vào của hệ thống giám sát qua Video là các luồng Video từ một hoặc nhiều Camera. Hệ thống sẽ phân tích nội dung của Video bằng cách tách Foreground ra khỏi Background, phát hiện và theo dõi đối tượng, và thực hiện phân tích thông tin ở mức cao, sau đó đưa ra kết quả theo yêu cầu của hệ thống.

Một hệ thống Video giám sát được mô tả như sau:

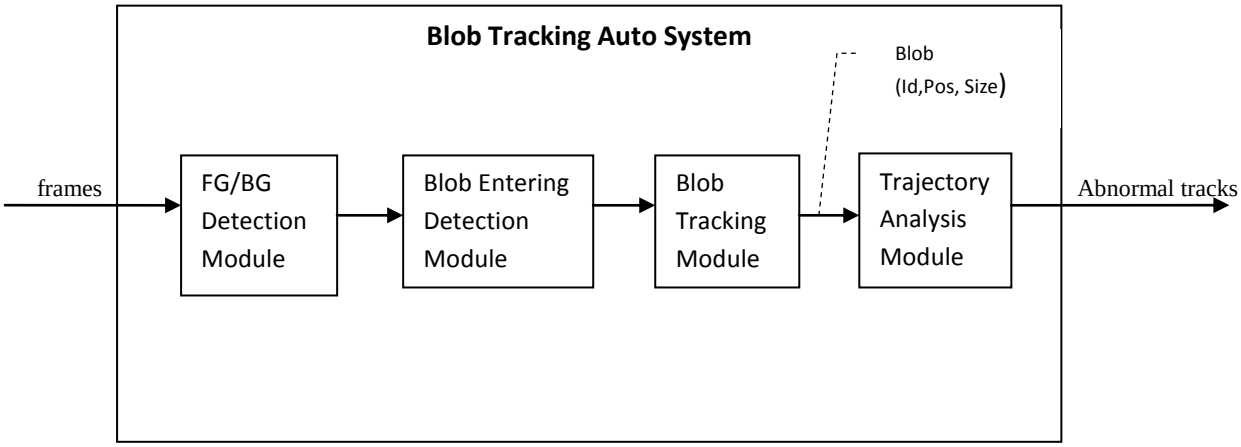
- A "Foreground/Background (FG/BG) Detection" module thực hiện phân loại FG/BG với từng pixel.

- A "Blob Entering Detection" module sử dụng kết quả của FG/BG Detection

Module phát hiện những đối tượng vào vùng giám sát.

- A "Blob Tracking" module được khởi tạo bởi "Blob Entering Detection" module. Module này theo dõi các đối tượng có trong danh sách các đối tượng mà "Blob Entering Detection" module cung cấp.

- A "Trajectory Analysis" module thực hiện phân tích quỹ đạo chuyển động của đối tượng và chỉ ra chuyển động nào bất thường.



Ước lượng Foreground và Background

Việc tách Foreground ra khỏi Background là bước quan trọng nhất trọng trong hệ thống Video surveillance. FG estimation là bước đầu tiên của hệ thống. Độ chính xác của bước này quyết định đến độ chính xác của các bước tiếp theo

của hệ thống và nó cúng ảnh hướng lớn đến thời gian thực thi của các bước tiếp theo.

Tách FG sẽ hoạt động tốt nhất trong điều kiện trong nhà, tuy nhiên chúng ta muốn có một hệ thống làm việc tốt cả trong nhà và ngoài trời. Môi trường bên ngoài phức tạp hơn trong nhà rất nhiều ví dụ như: chuyển động của cách cây, mặt nước gợn sóng… sẽ gây ra sai lầm. Thuật toán chúng ta sử dụng để tách FG là:

[14] *L. Li, W. Huang, I.Y.H Gu, Q. Tian, "Foreground object detection from videos containing complex background," ACM Multimedia 2003.*

[15] C. *Stauffer C, W.E.L Grimson, "Adaptive background mixture models fore real-time tracking," 1999 IEEE CVPR*

Blob Entering Detection

Blob entering detection module dựa trên việc theo dõi thành phần liên thông (connected component) được thực hiên như sau:

- Tính toán thành phần liên thông dựa trên kết quả thu được từ FG/BG estimation module. Mỗi thành phần này gọi là một *blob*.

- Theo dõi các blob này bằng cách tìm kiếm chúng trong frame hiện tại và quá khứ.

- Thêm blob mới vào danh sách các bob được theo dõi.

Tài liệu tham khảo:

[16] *A. Senior, A. hampapur, Y. L. Tian, L. Brown, S. Pankanti, R. Bolle, "Appearance models for occlusion handling," in proccedings of Second International workshop Performance Evaluation of Tracking and Surveillance systems in conjunction with CVPR'01, December 2001.*

Blob Tracking

Blob Tracking module cung cấp vị trí, kích thước của các blob đang được theo dõi. Module này được thiết kế gồm 2 thành phần. Thành phần thứ nhất có nhiệm vụ theo dõi các blob, thành phần thứ 2 được xây dựng dựa trên thuật toán *Mean-shift* và *particle filtering*. Thuật toán Kalman filter cũng được sử dụng để dự đoán vị của blob trong frame tiếp theo.

Tham khảo:

[17] D.Comaniciu, V. Ramesh, P. Meer, "Real-time tracking of non-rigid objects using mean shift," IEEE International Conference on Pattern Recognition, vol. 2, pp. 142-149, 13-15 June, 2000.

[18] K. Nummiaro, E. Koller-Meier, and L. Van Gool, "A color based particle filter," in First International Workshop on Generative-Model-Based Vision, A.E.C.Pece, Ed., 2002.

Activity Analysis

Để có thể phát hiện được các đối tương bất thường trong vùng quan sát ta sẽ phân tích quỹ đạo chuyển động của các đối tượng. Để có thể làm được điều này ta lưu các đặc điểm của mỗi blob tại thời điểm t bằng một 5-D vector. Mỗi thành phần của vector là một đăc điểm của blob như: vị trí (x, y), vận tốc (v_x,v_y), trạng thái (s). Sau khi có được các quan sát về đặc điểm của các bob ta phân tích mẫu thống kê này, tùy theo từng hoàn cảnh mà ta có thể đưa ra các tiêu chuẩn để xét blob nào bất thường blob nào bình thường.

Tài liệu tham khảo:

[19] N. Johnson and D.C. Hogg, "Learning the distribution of object trajectories for event recognition," Image and Vision Computing, 14:609-615, 1996.

[20] N. Sumpter, A.J Bulpitt, "Learning spatio-temporal patterns for predicting object behaviour," Image and Vision Computing, 18(9), pages 697-704, 2000.

[21] J. Owens, A. Hunter, "Application of the self-organising Map to Trajectory Classification," Proc. Third IEEE Visual Surveillance Workshop, 1 July 2000, Dublin, ISBN 0-7695-0698-4, pp. 77-83.

[22] J. Lou, Q. Liu, and T. Tan, W. Hu, "Semantic interpretation of object activities in a surveillance system," Pattern nocognition 2002, Volume 3, 11-15 Aug, 2002, pp.777-780.

4. Giới thiệu hệ thống camera giám sát trong OpenCV.

Để hiểu hơn về thư viện OpenCV các bạn có thể tìm đọc cuốn *Learning OpenCV Library, Pulisher: O'Reilly.*

Mô tả các cấu trúc dữ liệu cơ bản

CvBlob

Cấu trúc này nhằm mô tả vị trí, kích thước, ID của blob.

```
typedef struct CvBlob

{

    float    x,y; /* blob position    */

    float    w,h; /* blob sizes       */

    int      ID;  /* blbo ID          */

 }CvBlob;
```

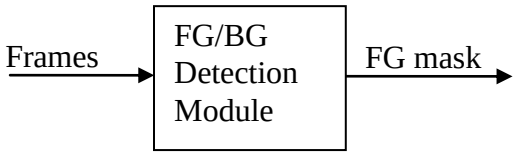
CvBlobList

Đây là một class lưu lại tập hợp các blob. Tập hợp này có thể là quỹ đạo các blob hay là danh sách các blob có trong frame hiện tại. Danh sách này được đánh thứ tự do đó có thể truy cập qua ID.

```
class CvBlobSeq

{

public:

    CvBlobSeq(int BlobSize = sizeof(CvBlob));

    virtual ~CvBlobSeq();

    virtual CvBlob* GetBlob(int BlobIndex);

    virtual CvBlob* GetBlobByID(int BlobID);

    virtual void DelBlob(int BlobIndex);

    virtual void DelBlobByID(int BlobID);

    virtual void Clear();

    virtual void AddBlob(CvBlob* pB);

    virtual int GetBlobNum();

};
```

Mô tả các Module

CvFGDetector



Đây là một virtual class, mô tả interface của "FG/BG Detection" module. Nếu ai đó muốn tạo một module của riêng mình thì phải tạo một class được thừa kế từ CvFGDector và implement tất cả các virtual method được khai báo trong CvFGDector.

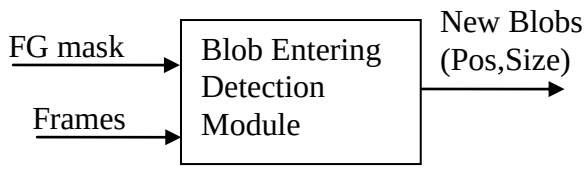
Input Data: frame hiện thời.

Output Data: FG/BG mask của frame hiện thời.

```
class CvFGDetector

{

public:

    virtual IplImage* GetMask() = 0;

    /* process current image */

    virtual void    Process(IplImage* pImg) = 0;

    /* release foreground detector */

    virtual void    Release() = 0;

};
```

Module này phân loại các pixel trên frame hiện thời thành FG hoặc BG pixel. Kết quả thu được là FG mask đươc trả về bởi GetMask method. Tất cả các method này phải được định nghĩa lại trong class con.

CvBlobDetector



Đây là một virtual class, mô ta interface của "Blob Entering Detection" module. Nếu ai đó muốn tạo riêng mình một module thì phải tạo một class được thừa kế

từ CvBlobDector class và implement tất cả các virtual method được khai báo trong CvBlobDector.

Input Data:

FG/BG mask của frame hiện tại.

Danh sách các blob đã tồn tại.

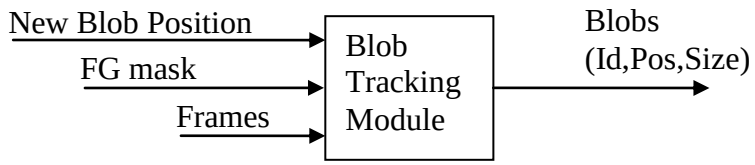
Output Data:

Danh sách các blob mới được phát hiện.

```
class CvBlobDetector

{

public:

    /* try to detect new blob entrance based on foreground mask */

    /* pFGMask - image of foreground mask */

    /* pNewBlobList - pointer to sequence to save new detected blobs  */

    /* pOldBlobList - pointer to blob list which already exist on image */

    virtual   int   DetectNewBlob(IplImage*   pFGMask,   CvBlobSeq*   pNewBlobList,
CvBlobSeq* pOldBlobList) = 0;

       /* return number of detected blobs */

    /* release blob detector */

    virtual void Release()=0;

};
```

Module này phát hiện các blob mới bằng cách sử dụng FG mask và danh sách các blob mới được lưu vào pNewBlobList. Module này cũng sử dụng danh sách các blob đã tồn tại trong pOldBlobList để nhận biết các blob mới. Tất cả các method phải được định nghĩa lại trong class con.

CvBlobTracker



Đây là một virtual class, mô tả interface của “Blob Tracking” module. Nếu bạn muốn tạo một module của mình thì phải tạo một class được thừa kế từ

CvBlobTracker và phải implement tất cả các virtual method có trong CvBlobTracker.

Input Data:

Frame hiện thời

FG/BG mask của frame hiện thời.

Output Data:

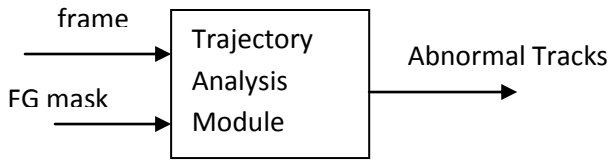
Blobs(Id, pos, size) trên frame hiện thời.

```
class CvBlobTracker

{

public:

    /* Add new blob to track it and assign to this blob personal ID */

    /* pBlob - pinter to structure with blob parameters (ID is ignored)*/

    /* pImg - current image */

    /* pImgFG - current foreground mask */

    /* return pointer to new added blob */

    virtual CvBlob* AddBlob(CvBlob* pBlob, IplImage* pImg, IplImage* pImgFG = NULL
) = 0;

    /* return number of currently tracked blobs */

    virtual int     GetBlobNum() = 0;

    /* return pointer to specified by index blob */

    virtual CvBlob* GetBlob(int BlobIndex) = 0;

    /* delete blob by its index */

    virtual void    DelBlob(int BlobIndex) = 0;

    /* process current image and track all existed blobs */

    virtual void    Process(IplImage* pImg, IplImage* pImgFG = NULL) = 0;

    /* release blob tracker */

    virtual void    Release() = 0;


    /* return pinter to blob by its unique ID */

    virtual int     GetBlobIndexByID(int BlobID);

    /* return pinter to blob by its unique ID */
```

```
    virtual CvBlob* GetBlobByID(int BlobID);

    /* delete blob by its ID */

    virtual void    DelBlobByID(int BlobID);

    /* Set new parameters for specified (by index) blob */

    virtual void    SetBlob(int BlobIndex, CvBlob* pBlob);

    /* Set new parameters for specified (by ID) blob */

    virtual void    SetBlobByID(int BlobID, CvBlob* pBlob);

};
```

Module này chứa danh sách các blob mà danh sách này sẽ được update sau khi Process() được gọi. Người dùng có thể update các blob thủ công bằng các method SetBlob và SetBlobByID. Các method AddBlob, GetBlobNum, DelBlob, Process, Release phải được định nghĩa lại trong class con, các method khác có thể định nghĩa lại nếu cần thiết.

CvBlobTrackAnalysis



Đây là một virtual class mô tả interface của *"Trajectory Analysis"* module, nếu bạn muốn tạo cho mình môt module riêng thì bạn phải tạo một class được thừa kế từ CvBlobTrackAnalysis, và bạn phải implement toàn bộ các virtual method có trong class.

Input Data:

Frame hiên thời,

FG mask

Output Data:

Blob là bình thường hay bất thường.

```
class CV_EXPORTS CvBlobTrackAnalysis: public CvVSModule
{
public:
       CvBlobTrackAnalysis(){SetTypeName("BlobTrackAnalysis");};
    virtual void    AddBlob(CvBlob* pBlob) = 0;
    virtual void    Process(IplImage* pImg, IplImage* pFG) = 0;
    virtual float   GetState(int BlobID) = 0;
```

```
    /* return 0 if trajectory is normal
       return >0 if trajectory abnormal */
    virtual const char*   GetStateDesc(int /*BlobID*/){return NULL;};
    virtual void    SetFileName(char* /*DataBaseName*/){};
    virtual void    Release() = 0;
};
```

Process method sẽ update trạng thái của các blob có trong frame sau khi được gọi. GetState method trả lại 0 nếu blob là bình thường, lớn hơn 0 nếu blob là bất thường.

5. Kết luận.

"Recognition", "Minning", "Synthesis" (RMS) sẽ là trọng tâm nghiên cứu của *"Data processing"* trong tương lai. Lĩnh vực CV sẽ là lĩnh vực quan trọng của RMS. Qua bài viết này chúng ta thấy được nhưng xu thế ứng dụng và xu thế nghiên cứu trong CV. Sau đó chúng ta tìm hiểu về hệ thống Video giám sát. Video giám sát là một trong những ứng dụng quan trọng nhất trong CV. Hy vọng rằng sau khi đọc bài này các bạn có thêm những ý tưởng nhằm cải tiến hệ thống hoạt động hiệu quả hơn.

Mọi câu hỏi và ý kiến đóng góp xin gửi về:

Email: duytung88@gmail.com

YM: spider_fx01